Department of Computer Engineering

Academic Term: First Term 2023-24

Class: T.E /Computer Sem – V / Software Engineering

Practical No:	2
Title:	**Software Requirement Specification**
Date of Performance:	**02 / 08 / 2023**
Roll No:	**9542**
Team Members:	**Group 8**

Rubrics for Evaluation:

Sr. No	Performance Indicator	Excellent	Good	Below Average	Total Score
1	On time Completion & Submission (01)	01 (On Time)	NA	00 (Not on Time)	
2	Theory Understanding(02)	02(Correct)	NA	01 (Tried)	
3	Content Quality (03)	03(All used)	02 (Partial)	01 (rarely followed)	
4	Post Lab Questions (04)	04(done well)	3 (Partially Correct)	2(submitted)	

Signature of the Teacher:

Department of Computer Engineering

Academic Term: First Term 2022-23

Class: T.E /Computer Sem – V / Software Engineering

Signature of the Teacher:

Lab Experiment 02

Experiment Name: Implementing Project Using Scrum Method on JIRA Tool in Software Engineering

Objective: The objective of this lab experiment is to introduce students to the Scrum framework and its implementation using the JIRA tool. Students will gain practical experience in managing a software project using Scrum principles and learn how to utilize JIRA as a project management tool to track and organize tasks, sprints, and team collaboration.

Introduction: Scrum is an agile project management methodology that promotes iterative development, collaboration, and continuous improvement. JIRA is a widely used tool that supports Scrum practices, providing teams with features to plan, track, and manage software projects effectively.

Lab Experiment Overview:

1. Introduction to Scrum: The lab session begins with an overview of the Scrum framework, including its roles (Product Owner, Scrum Master, and Development Team), events (Sprint Planning, Daily Standup, Sprint Review, and Sprint Retrospective), and artifacts (Product Backlog, Sprint Backlog, and Increment).
2. JIRA Tool Introduction: Students are introduced to the JIRA tool and its capabilities in supporting Scrum project management. They learn to create projects, epics, user stories, tasks, and sub-tasks in JIRA.
3. Defining the Project: Students are assigned a sample software project and create a Product Backlog, listing all the required features, user stories, and tasks for the project. 4. Sprint Planning: Students organize the Product Backlog into Sprints, selecting user stories and tasks for the first Sprint. They estimate the effort required for each task using story points. 5. Implementation in JIRA: Students use the JIRA tool to create a Sprint Backlog, add the selected user stories and tasks, and assign them to team members.
6. Daily Standup: Students conduct a simulated Daily Standup meeting, where they update the progress of their tasks and discuss any impediments they are facing.
7. Sprint Review and Retrospective: At the end of the Sprint, students review the completed tasks, demonstrate the implemented features, and gather feedback from their peers. They also conduct a Sprint Retrospective to identify areas of improvement for the next Sprint.
8. Continuous Iteration: Students continue implementing subsequent Sprints, repeating the Sprint Planning, Daily Standup, and Sprint Review & Retrospective events.
9. Conclusion and Reflection: At the end of the lab experiment, students reflect on their experience with Scrum and JIRA, discussing the advantages and challenges they encountered during the project.

Learning Outcomes: By the end of this lab experiment, students are expected to:

- Understand the Scrum framework and its principles in agile project management.
- Gain practical experience in using the JIRA tool for project management in a Scrum environment. · Learn to create and manage Product Backlogs, Sprint Backlogs, and track progress using JIRA. · Develop collaborative skills through Daily Standup meetings and Sprint Reviews. · Gain insights into the iterative nature of software development and the importance of continuous improvement.

Pre-Lab Preparations: Before the lab session, students should familiarize themselves with the Scrum framework and the basics of the JIRA tool. They should review Scrum roles, events, and artifacts, as well as the features of JIRA relevant to Scrum implementation.

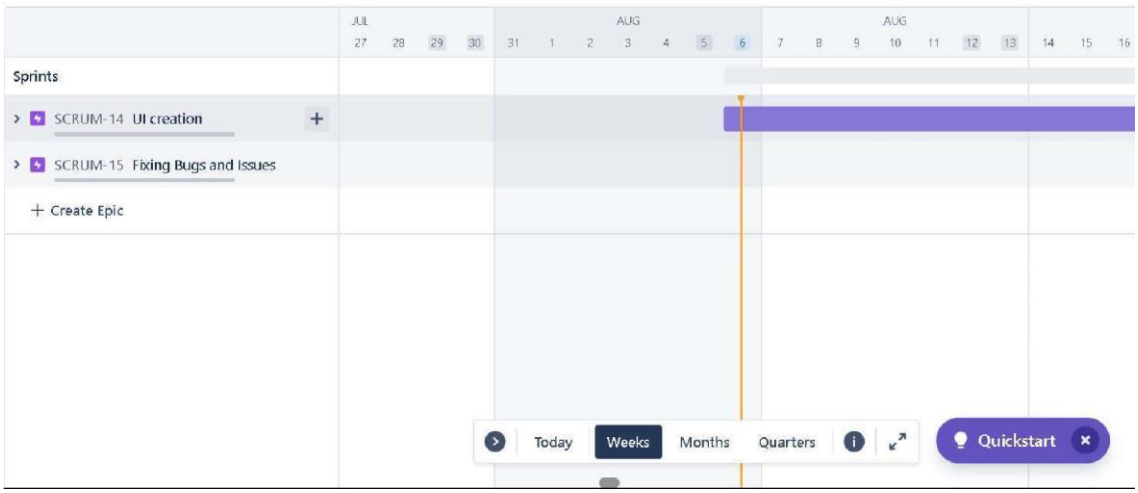
Materials and Resources:

- Computers with internet access for accessing the JIRA tool
- Project brief and details for the sample software project
- Whiteboard or projector for explaining Scrum concepts

Conclusion: The lab experiment on implementing a project using Scrum on the JIRA tool offers students hands-on experience in agile project management. By utilizing Scrum principles and JIRA's capabilities, students learn to collaborate effectively, manage tasks efficiently, and adapt to changing requirements. The practical exposure to Scrum and JIRA enhances their understanding of agile methodologies, equipping them with valuable skills for real-world software development projects. The lab experiment encourages students to embrace the agile mindset, promoting continuous improvement and customer-centric software development practices.

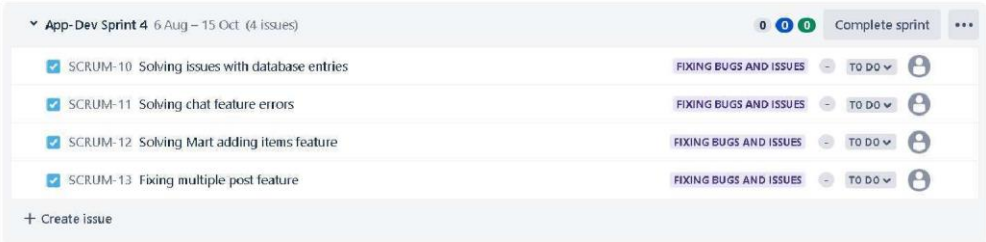
Outpt

EPIC

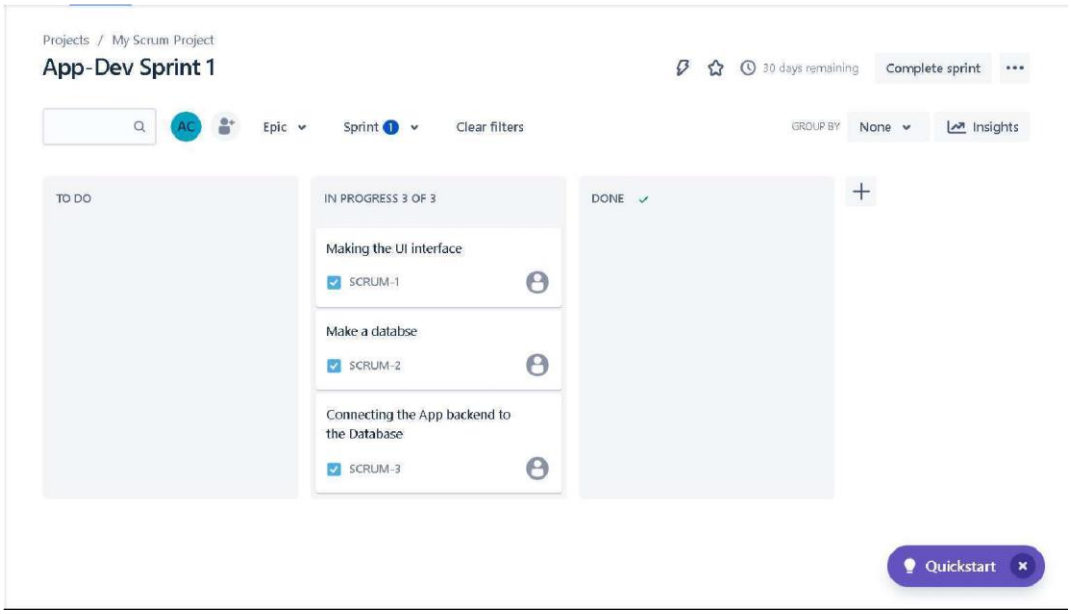


Backlog

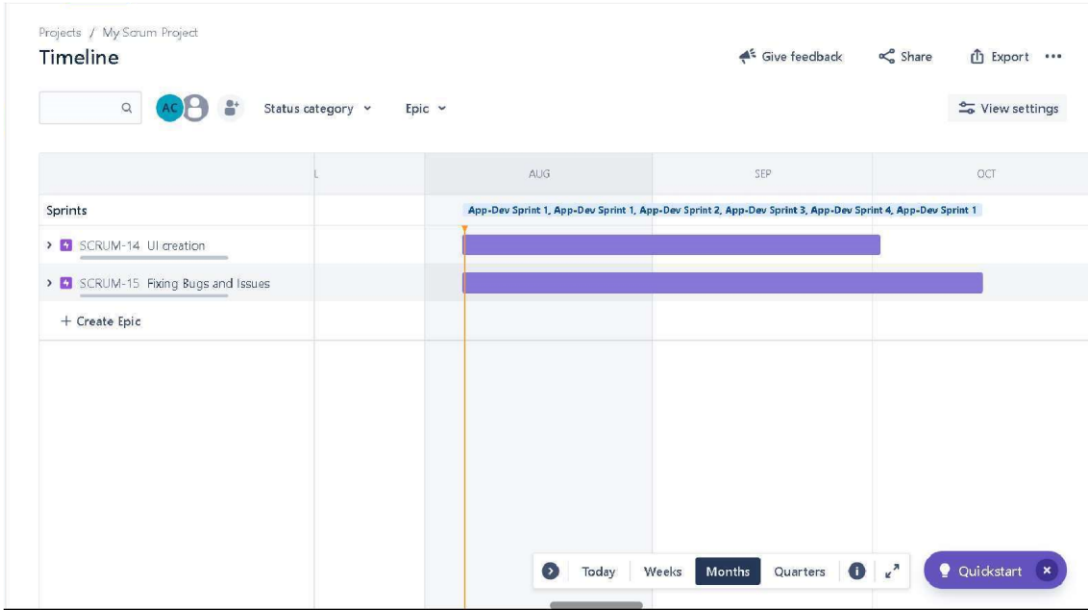
Sprint



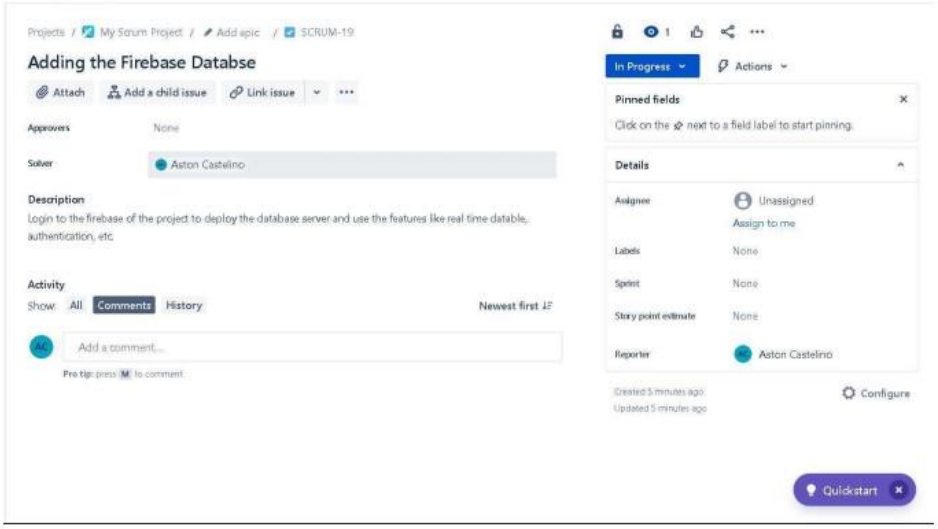
Sprint Board



Timeline



Tasks


Projects / My Scrum Project / Add epic / SCRUM-19
Adding the Firebase Databse
Attach
Add a child issue
Link issue
Approvers
None
Solver
Aston Castelino
Description
Login to the firebase of the project to deploy the database server and use the features like real time datable, authentication, etc
Activity
Show: All Comments History
Newest first
Add a comment...
Pro tip: press M to comment
In Progress
Actions
Pinned fields
Click on the next to a field label to start pinning.
Details
Assignee
Unassigned
Assign to me
Labels
None
Sprint
None
Story point estimate
None
Reporter
Aston Castelino
Created 5 minutes ago
Updated 5 minutes ago
Configure
Quickstart

1 . Scrum framework

The Scrum framework is a popular Agile methodology used for managing complex projects. It emphasizes iterative and incremental development, frequent inspection and adaptation, and collaboration within cross-functional teams. Here are some of the advantages and disadvantages of the Scrum framework:

Advantages:

1. Flexibility and Adaptability: Scrum allows teams to respond quickly to changing requirements and priorities. The iterative nature of Scrum allows for continuous refinement and adjustment based on feedback, ensuring that the product remains aligned with customer needs.

2. Increased Collaboration: Scrum promotes close collaboration between team members, stakeholders, and customers. Daily stand-up meetings, sprint planning, and retrospective meetings facilitate communication, transparency, and teamwork.

3. Frequent Deliverables: Scrum divides work into time-boxed iterations called sprints, typically lasting two to four weeks. This results in frequent deliverables, allowing stakeholders to see tangible progress and provide feedback, which can lead to early course corrections.

4. Customer Satisfaction: Scrum focuses on delivering value to customers early and regularly. This can lead to improved customer satisfaction as they see their needs being addressed incrementally throughout the project.

5. Continuous Improvement: Regular retrospective meetings provide a platform for the team to reflect on their processes and identify areas for improvement. This culture of continuous improvement helps teams refine their practices and become more efficient over time.

6. Transparency: Scrum promotes transparency by requiring that work, progress, and impediments be visible to all team members. This reduces misunderstandings and encourages open communication.

Disadvantages:

1. Complexity: Implementing Scrum effectively requires a deep understanding of its principles and practices. It can be challenging for teams new to Agile methodologies to grasp and follow the framework correctly.

2. Resistance to Change: Traditional organizations might resist the shift to Agile practices, as Scrum requires changes in team roles, communication patterns, and management styles.

3. Lack of Detailed Documentation: Scrum emphasizes working software over comprehensive documentation. While this is efficient for some projects, it might not align well with industries or projects that require extensive documentation for compliance or other reasons.

4. Potential for Overcommitment: Inaccurate sprint planning can lead to teams overcommitting to work, resulting in stress, burnout, and compromised quality.

5. Dependency on Strong Team Collaboration: Scrum relies heavily on collaboration among team members. If there are issues with teamwork, communication, or conflicts, it can negatively impact the effectiveness of the framework.

6. Short-Term Focus: The iterative nature of Scrum can sometimes lead to a focus on short-term goals and neglecting long-term architectural or infrastructure concerns.

7. May Not Suit All Projects: Scrum is best suited for projects with changing requirements and a need for rapid delivery. It might not be the optimal choice for projects with very rigid or predictable requirements.

.

2 . Traditional project management

Traditional project management refers to methodologies that follow a linear and structured approach to project planning, execution, and monitoring. One of the most well-known traditional project management methodologies is the Waterfall model. Here are some advantages and disadvantages of traditional project management:

Advantages:

1. Clear Project Scope and Requirements: Traditional project management methods emphasize detailed upfront planning, including defining the project scope and requirements. This can lead to a clear understanding of what needs to be accomplished.

2. Predictability: Traditional methods are often well-suited for projects with stable and well-defined requirements. The linear nature of these methods can lead to a sense of predictability in terms of timelines and outcomes.

3. Structured Approach: Traditional methodologies provide a clear and structured sequence of phases, such as requirements gathering, design, development, testing, and

deployment. This can help ensure that each phase is completed before moving on to the next.

4. Comprehensive Documentation: Traditional approaches typically require comprehensive documentation at each phase of the project. This documentation can serve as a reference for future projects, audits, and compliance purposes.

5. Appropriate for Well-Defined Projects: Traditional methods work well for projects where requirements are fixed, well-understood, and unlikely to change significantly throughout the project lifecycle.

Disadvantages:

1. Rigidity: Traditional methods are often inflexible when it comes to adapting to changes in requirements, scope, or priorities. This can lead to challenges when dealing with evolving customer needs or market conditions.

2. Limited Customer Involvement: Traditional approaches prioritize upfront planning over customer involvement. This can result in projects that do not fully meet customer needs or expectations.

3. Late Feedback: Because traditional methodologies typically involve completing each phase before moving on to the next, feedback from stakeholders often occurs late in the project lifecycle. This can make it challenging to address issues or changes effectively.

4. Long Time to First Deliverable: Traditional methods can result in a long time between project initiation and the first tangible deliverable. This delay can impact the project's ability to respond to changing market conditions or customer needs.

5. Costly Changes: Making changes to project requirements or scope after the project has started can be costly and time-consuming in traditional approaches, as it may require revisiting earlier phases.

6. Risk of Scope Creep: Due to the limited involvement of stakeholders and customers during development, there is a risk of scope creep—uncontrolled expansion of project scope—which can lead to delays and budget overruns.

7. High Documentation Overhead: The emphasis on comprehensive documentation at each phase can result in high administrative overhead and slow down the project's progress.

Effectiveness Comparison:

Scrum is more effective when:

- Requirements are uncertain or evolving.
- Collaboration and stakeholder involvement are crucial.
- Quick value delivery and continuous improvement are important.
- The project requires flexibility and adaptability.

Traditional methods are more effective when:

- Requirements are well-defined and unlikely to change.
- Comprehensive documentation is necessary.
- Predictability and strict control are priorities.
- The project is in a highly regulated industry.

b.)

Analysis of Potential Bottlenecks or Issues from Experiment 1 Project in JIRA:

Uneven Task Distribution: The task distribution is uneven, with some team members having multiple tasks while others have none. This might lead to a lack of balance in workloads and delays in completing specific features.
Dependency on Single Team Members: If one team member (e.g., Sarah) is assigned to critical tasks across different user stories, the team's progress could be affected if that person faces any delays or issues.
Inadequate Testing Coverage: There are no tasks specifically assigned for testing user stories. This could lead to inadequate testing, reduced quality, and potential issues later in the sprint.
Unstarted Tasks: Several tasks are marked as "Not Started." If these tasks are not initiated promptly, they might become bottlenecks, delaying the completion of user stories.

Lack of Task Breakdown: Some user stories (e.g., "Pet Listing") might require more detailed task breakdowns to ensure a clear understanding of what needs to be accomplished.
Missing Acceptance Criteria: If the user stories lack well-defined acceptance criteria, it could lead to misunderstandings about when a user story is truly complete.
Communication and Coordination: The team might face communication challenges if there's a lack of coordination among team members, especially if they're working on interdependent tasks.

c.)

The Scrum Master plays a critical role in handling conflicts within the development team and resolving impediments to maintain a smooth project flow in a Scrum framework. Their responsibilities encompass fostering a collaborative and productive environment while ensuring that the team can effectively follow the Scrum methodology. Here's an evaluation of the Scrum Master's role in conflict resolution and impediment removal:

Conflict Resolution:

Observation and Detection: The Scrum Master actively monitors team dynamics, communication patterns, and behaviors to identify any signs of conflict. This includes both explicit disputes and underlying tensions.
Facilitation: When conflicts arise, the Scrum Master facilitates open discussions among team members, allowing everyone to express their perspectives and concerns. They create a safe space for communication and ensure that all voices are heard.
Mediation: In situations where conflicts escalate, the Scrum Master acts as a neutral mediator, helping team members find common ground and guiding them toward mutually beneficial solutions.
Coaching and Empowerment: The Scrum Master coaches team members in conflict resolution techniques and helps them develop effective communication skills. They empower the team to address conflicts independently.
Conflict Prevention: The Scrum Master proactively works to prevent conflicts by encouraging transparency, fostering a culture of respect, and addressing issues before they escalate.

Impediment Removal:

Identifying Impediments: The Scrum Master identifies obstacles, bottlenecks, and impediments that hinder the team's progress. These can be related to processes, tools, communication, or external factors.
Prioritization: The Scrum Master helps the team prioritize impediments based on their impact on project progress and sprint goals.